THE ISLAMIC UNIVERSITY – FACULTY OF ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

LCOM 3010 – Introduction to Computer Architecture Lab

Lab #4
REGISTER FILE DESIGN

BY

Eng: Rasha Ghazy Sammour

Submitted for

Dr. Mohammed Mikki

March 31, 2021

Register File Design

Objective:

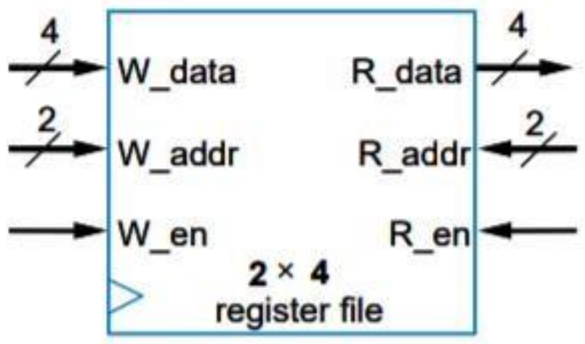
1. Design of register files, synthesis and implementation.
2. Emulation of register file performance

Theory: Design of Register File:

A typical data path has multiple registers. The construction of a bus system with a large number of registers requires different techniques. A set of registers having common operations performed on them maybe organized into a register file*.

A block diagram of a 2x4 register file is shown in Figure L7-1. Note that in general a register file is denoted as 2m x n, where m is the number of register address bits and n is the number of bits per register. In our case it is better to use a notation 22 x 4 register file†.

Figure L7-1. Block Diagram of a 2x4 Register File

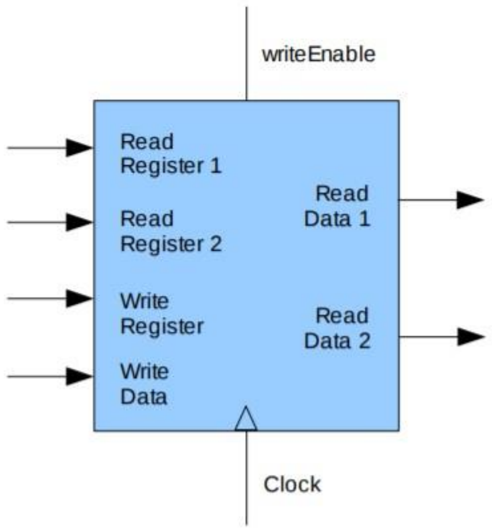


Register files have a memory-like behavior. However, they are very different from memories like SRAM, ROM, etc. as will be explained in more detail in class. They are generally used as fast temporary storage. Brief comparative properties are outlined in Table L7-1 below

Register File	Memory (SRAM)
m n-bit storage words	*m n*-bit storage words
Few words (<256), many ports, dedicated read/write ports, so that read and write operations can be done simultaneously.	Many words (>1024), few ports, shared read/write ports. Read and write operations cannot be done simultaneously.
Writing to a register takes just one clock cycle. Read op doesn't require a clock or additional control inputs.	Writing to a register can take many clock cycles. Read op requires a clock and other CTRL.
Logically Static Content	Logically Dynamic Content
Synchronous	Different implementations but mostly asynchronous‡

Table L7-1. Register Files vs. SRAM Memory

A register file is a sub circuit that lets the surrounding circuitry read from one or more registers. A register file also lets the surrounding circuitry optionally write to one or more registers. In this part of the lab, you will build a register file that has two read ports and one write port. A register file always outputs the contents of the registers corresponding to the Read Register inputs. In addition, the value of the register corresponding to the Write Register input is updated with the value of the Write Data input when the Write Enable signal is activated and there is a rising clock edge.



A register file can be implemented with a multiplexer for each read port, a decoder for the write port and an array of registers built from D flip-flops. Implement a register file in Logisim. The register file should have four 16-bit registers, two read ports and one write port.

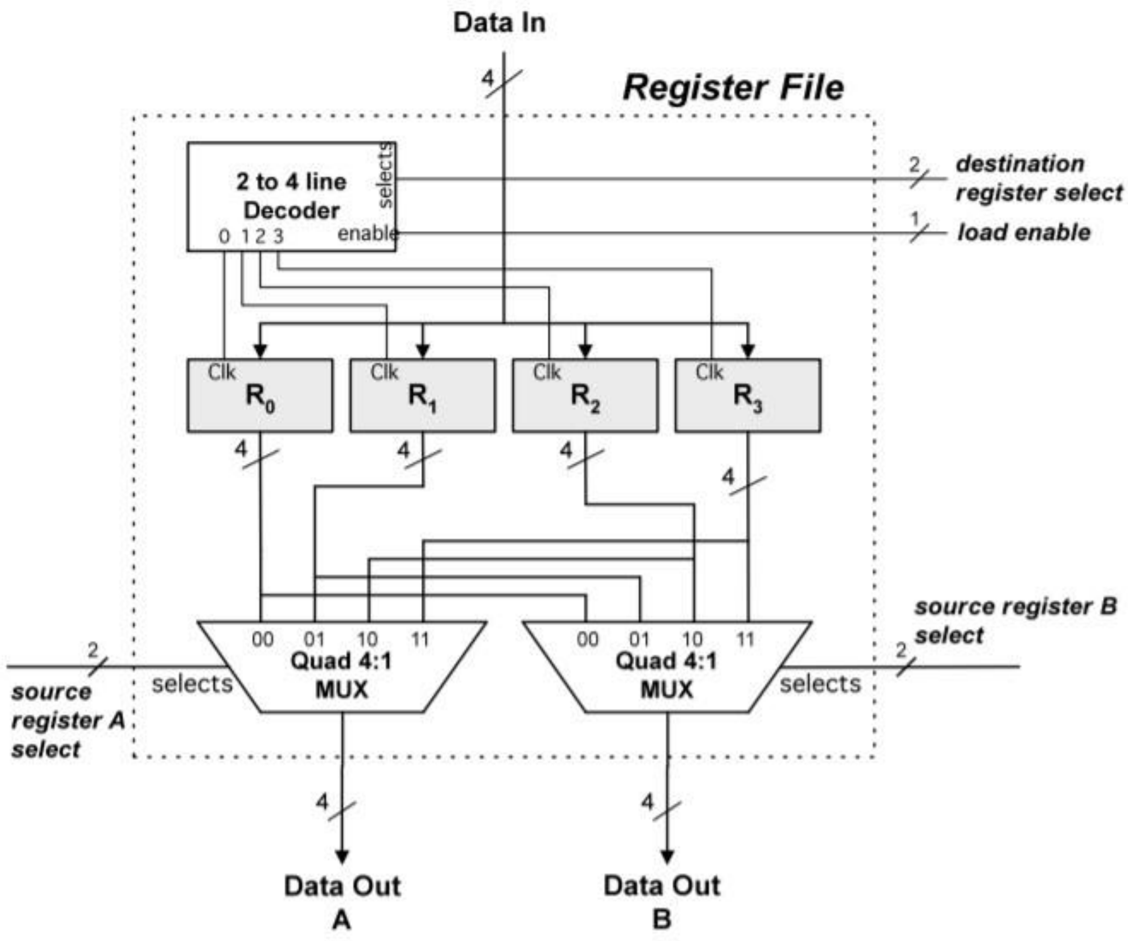
Input/output descriptions:

- ReadRegister1 specifies which register should be read from. That register's data should be presented on the ReadData1 output.
- ReadRegister2 specifies which register should be read from. That register's data should be presented on the ReadData2 output.
- WriteRegister specifies which register should be written to.
- WriteData is the data that will be written to the register specified by WriteRegister.

On a rising clock edge, the data will be written to the specified register.

- WriteEnable, if true, will cause the WriteData to be written to the register specified by Write-Register; otherwise, WriteData will be ignored and no register will be written to. ● Clock conveys the clock signal.

Here is a picture of what your register file sub circuit, symbolically, looks like. Bold labels indicate that an input/output contains more than 1 bit of data.



Assignment Statements:

1. Design a 32 Register File.
2. Connect Register File in previous problem with ALU that designed in Lab2.